14

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/084,780	02/25/2002	Christopher Andrew Hinsley	4362-4002	1272

27123 7590 03/22/2007
MORGAN & FINNEGAN, L.L.P.
3 WORLD FINANCIAL CENTER
NEW YORK, NY 10281-2101

EXAMINER	
VO, TED T	
ART UNIT	PAPER NUMBER
2191	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	03/22/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No. 10/084,780	Applicant(s) HINSLEY ET AL.
	Examiner Ted T. Vo	Art Unit 2191

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *21 December 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-11 and 13-30* is/are pending in the application.

4a) Of the above claim(s) _-_ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-11 and 13-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the amendment filed on 12/21/2006.

Claims 1-11, 13-30 are pending in the application.

Specification

2. In the specification, cross-reference should identify its related application and use the root words incorporation by reference. Applicants fail to include in accordance to 37 CFR 1.57.

Response to Arguments

3. In view of the Amendment, the rejection under the second paragraph of 35 USC 112 is withdrawn.

Regarding the arguments to the claims under claim rejection 35 USC 103.

It appears Applicants do not discuss what is the novelty of the claim or point out the patentable feature in the claim, but the argument remains the same as in the previous reply. For example, Applicants in the remarks of p. 12 contends the compilation to the native code of the reference occurs at one and the same place as the translation to the intermediate language (another argument, "internal", "they are clearly not transmitted in the network", remarks: p. 14). Examiner disagreed. The Figure 2 is a scheme. At p. 15, the reference clearly discusses the transmission of neutral code, IL code, used by IBM. IBM clearly sent IL that is as output of byte code to its VM (See p. 15). It should be noted that at the time the applicants' filing, the compilation of code at one computer then transmitting the compiled code into other computer is not new. The byte code it self is a neutral code. Compiler is a portable device; it can place everywhere. If Applicants believe that they are the first to do this, then Examiner would like to hear their discussion. It appears the argument contends that sending neutral code generated from bytecode in a network is novel. Examiner contends that code sent from a computer to another computer is in public use and old. The reference also mentions the Sun had performed compiling Java source code into Java bytecode, and sent the code to a client target JVM (See p. 2 second paragraph, "Java source code is translated into architecture neutral bytecodes that can be executed on any platform". As mentioned above, p. 15, the reference shows front end compiler such as HPCJ that even Java source code or byte code as input and coverts it into IL. Another backend compiler from IBM's XL translates IL into object module. The reference shows that the every kind of code could be sent from a server to every user's JVM. In the Figure 2, it does not explicitly address the transmission as between servers and clients. However, the scheme of Figure 2 reads the claim if one know that the "arrows" in the figure could represent the network transmission lines.

More importantly, the reference teaches the heart of the claims, converting "Bytecode" into "Intermediate Language", a form of neutral code before it is translated into a native code. Thus, bytecode, a kind of neutral code, or converting "Byte code" into another "neutral language" is already a prior art.

Therefore, transmitting in *transmitting the virtual processor code from a server to a client device"* is only to take whatever of communication technology, and thus it belongs to the public domain.

Analyzing the claim:

A method of translating an object-oriented computer program comprising:

(a) translating the program bytecode into machine independent virtual processor code which uses an instruction set of a virtual processor (See Byte code Translator, Figure 2, p. 7);

>>Examiner contends: It is known in the art.

(b) transmitting the virtual processor code from a server to a client device: *and*

>>Examiner contends: It is known in the art. It depends on the perspective view of the Figure 2. For this perspective view, it covers remote or within. A claim would be invalidated if it cannot be exclusive.

(c) translating the virtual processor code into native code which uses an instruction set of a physical processor of the client device (See Native Machine code, Figure 2),

>>Examiner contends: It is known in the art.

wherein the bytecode is stack-based, and in which the virtual processor code is register-based

>>Examiner contends: The recitation is not what it does, but "What it is"; where bytecode and JVM given in Kazi' reference includes "what it is", i.e., JVM is a stack-based machine – JVM includes registers, stacks, garbage-collected heap, methods areas, and execution engine.

It should be noted that the "non-obviousness" regards only to the lack in suggestion or the motivation; i.e., it is applied **"only"** to **an inventive feature**. The non-discussion in a reference if it is not an inventive feature does not need the this type of thing, "suggestion", because,

> MPEP 706.02(j): "To support the conclusion that the claimed invention is directed to obvious subject matter, either the references must expressly or impliedly suggest the **<u>claimed invention</u>** or the **<u>examiner must present a convincing line of reasoning as to why the artisan would have found the claimed invention to have been obvious in light of the teachings of the references</u>**." Ex parte Clapp, 227 USPQ 972, 973 (Bd. Pat. App. & Inter. 1985). See MPEP § 2144 - § 2144.09 for examples of reasoning supporting obviousness rejections.

In the claiming feature, **"transmitting virtual processor code from a server to a client device"**, it is **<u>not an inventive feature</u>**. It is well known, i.e. every computer user knew it before the filing of this application, i.e. all users who used a network knew **how to receive data, such as byte code, applets, from a server.**

Regarding the arguments to the claims under claim rejection 35 USC 102, as noted that, Applicants' claiming appears to cover two well-known features in the system claim: "a server" and "plurality remote client devices". Applicant's arguments remain the same as they argued in the previous reply.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

5. Claims 19-25 are rejected under 35 U.S.C. 102(e) as being anticipated by Miller et al, U.S. Patent No. 6,389,590.

<u>As per claim 19</u>: Miller et al discloses a distributed computer system (col. 4 lines 32-37) comprising a server including a store for storing virtual processor code, said code being a machine-independent representation of the bytecode of an object oriented computer program using an instruction set of a virtual processor (col. 4 lines 32-37, col. 1 lines 10-15, col. 1 lines 15-20, col. 6 lines 1-10);
a plurality of remote client devices in communication with the server, each client device including a client processor, a native translator arranged to translate the virtual processor code into native code which uses the instruction set of the respective client processor, and a native code store; the system including transmission means for transmitting the virtual processor code from the server to the client devices (col. 4 lines 32-37, col. 1 lines 15-20, col. 6 lines 1-10 and Fig. 2).

<u>As per claim 20</u>, Miller further discloses distributed computer system as claimed in claim 19 in which the transmission means consists of or includes a wireless network (col. 5 lines 3045).

As per claim 21, Miller further discloses distributed computer system as claimed in claim 20 in which the client devices are mobile phones (col. 4 lines 26-32).

As per claim 22, Miller further discloses a distributed computer system as claimed in claim 20 in which the client devices are hand-held computers (col. 4 lines 26-32).

As per claim 23, Miller further discloses a distributed computer system as claimed in claim 19 which the client devices are hand-held games consoles (col. 4 lines 26-32).

As per claim 24, Miller further discloses a distributed computer system as claimed in claim 19 in which at least one of the client devices includes a first type of client processor and in which at least another of the client devices includes a second type of client processor, using a different instruction set from that of the first type (col. 1 lines 10-20, col. 6 lines 1-10 and Fig. 2).

As per claim 25, Miller further discloses which the server .is further arranged to translate the object-oriented computer program from bytecode into virtual processor code (col. 1 lines 1020, col. 6 lines 1-10 and Fig. 2).

6. Claims 19-25 are rejected under 35 U.S.C. 102(b) as being anticipated by Kazi et al, "Techniques for Obtaining High Performance in Java Program", 7-1999.

As per claim 19: Kazi discloses *A distributed computer system comprising a server including a store for storing virtual processor code, said code being a machine-independent representation of the bytecode of an object oriented computer program using an instruction set of a virtual processor* (See Figure 2, p. 7: either Byte code "*machine-independent representation of an object oriented* " or IL code);

and a plurality of remote client devices (See Figure 2, target processor: note the Figure shows generally a source to an arbitrary target) *in communication with the server, each client device including a client processor, a native translator arranged to translate the virtual processor code into native code which uses the instruction set of the respective client processor* (See Figure 2, any of Interpreter, JIT compiler, Direct Complier, or Bytecode Translator), *and a native code store* (See Figure 2, Native Machine Code);

the system including transmission means for transmitting the virtual processor code from the server to the client devices (The connection medium between the Java Source Code/Bytecode Translator and Target processor).

As per claim 20: Kazi discloses *The distributed computer system as claimed in claim 19 in which the transmission means consists of or includes a wireless network* (The connection medium between the Java Source Code/Bytecode Translator and Target processor).

As per claim 21: Kazi discloses *The distributed computer system as claimed in claim 20 in which the client devices are mobile phones* (See Figure 2: Target Processor).

As per claim 22: Kazi discloses *The distributed computer system as claimed in claim 20 in which the client devices are hand-held computers* (See Figure 2: Target Processor).

As per claim 23: Kazi discloses *The distributed computer system as claimed in claim 19 in which the client devices are hand-held games consoles* (See Figure 2: Target Processor).

As per claim 24: Kazi discloses *The distributed computer system as claimed in claim 19 in which at least one of the client devices includes a first type of client processor and in which at least another of the client devices includes a second type of client processor, using a different instruction set from that of the first type* (See Target Processor: a generic processor in an arbitrary device, see either bytecode transmitted into an interpreter of the target processor or IL of Bytecode Translator that is independent-platform code).

As per claim 25: Kazi discloses *The distributed computer system as claimed in any one of claims 19 in which the server is further arranged to translate the object-oriented computer program from bytecode into virtual processor code.* (See Figure 2, either Bytecode: *virtual processor code, or* the Bytecode Translator computer that generates IL: *virtual processor code*).

7. Claims 19-25 are rejected under 35 U.S.C. 102(b) as being anticipated by Koizumi et al, U.S. Patent No. 5,586,323.

As per claim 19: *A distributed computer system comprising*

a server including a store for storing virtual processor code, said code being a machine-independent representation of the bytecode of an object oriented computer program using an instruction set of a virtual processor (See col. 4:40-45 'single abstract object program', and the computer system that generates this program: server), *and a plurality of remote client devices in communication with the server, each client device including a client processor, a native translator arranged to translate the virtual processor code into native code which uses the instruction set of the respective client processor, and a native code store* (See col. 4:46-50 "an installer" that resides in Machine A/Machine B, and this installer translates 'single abstract object program' into machine language of its target: Machine A or Machine B);

the system including transmission means for transmitting the virtual processor code from the server to the client devices (The connection medium between the computer that generates 'single abstract object program' and Machine A/Machine B).

As per claim 20: Koizumi discloses *The distributed computer system as claimed in claim 19 in which the transmission means consists of or includes a wireless network* (The connection medium between the computer that generates 'single abstract object program' and Machine A/Machine B).

As per claim 21: Koizumi discloses *The distributed computer system as claimed in claim 20 in which the client devices are mobile phones* (See Machine A/Machine B).

As per claim 22: Koizumi discloses *The distributed computer system as claimed in claim 20 in which the client devices are hand-held computers* (See Machine A/Machine B).

As per claim 23: Koizumi discloses *The distributed computer system as claimed in claim 19 in which the client devices are hand-held games consoles* (See Machine A/Machine B).

As per claim 24: Koizumi discloses, *The distributed computer system as claimed in claim 19 in which at least one of the client devices includes a first type of client processor and in which at least another of the client devices includes a second type of client processor, using a different instruction set from that of the first type* (See Machine A and Machine B where A and B are different platform and 'single abstract object program' is independent platform code).

As per claim 25: Koizumi discloses *The distributed computer system as claimed in any one of claims 19 in which the server is further arranged to translate the object-oriented computer program from bytecode into virtual processor code.* (See FIG 2, the computer generates intermediate language from a source file into 'single abstract object program': *virtual processor code*).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 1-5, 13-18, 26-30 is rejected under 35 U.S.C. 103(a) as being unpatentable over Kazi et al, "Techniques for Obtaining High Performance in Java Program".

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per Claim 1: Kazi discloses:

A method of translating an object-oriented computer program comprising:

(a) translating the program bytecode into machine independent virtual processor code which uses an instruction set of a virtual processor (See Byte code Translator, Figure 2, p. 7);

(b) transmitting the virtual processor code from a server to a client device: and

(c) translating the virtual processor code into native code which uses an instruction set of a physical processor of the client device (See Native Machine code, Figure 2), *wherein the bytecode is stack-based, and in which the virtual processor code is register-based* (Kazi' reference: JVM is a stack-based machine – JVM includes registers, stacks, garbage-collected heap, methods areas, and execution engine).

Kazi does not explicitly address **in the Figure 2** "*(b) transmitting the virtual processor code from a server to a client device*", i.e., whether IL (interpreted as *virtual processor code*) is whether sent from a server or not. Kazi simply shows the IL code is sent to a Native Machine Code. However, in the designing concept, one simply implements a dawn line as a connection between two elements, transmitter and receiver. The concept of transmutation data is well known as a network, where a network simply provides at least a connection between a sending data source computer device such as a server and a receiving data device such as a client device/computer.

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to include networking, an available source, that has been used commonly, as a means for sending independent-platform intermediate code to another backend remotely for providing targeted translation because growing of many different processor-types. Doing so would conform to the common standard of processor execution.

As per Claim 2: Kazi further discloses, *The method as claimed in claim 1 in which the program bytecode includes a class file, the class file being converted into one or more virtual processor tools which use the instruction set of the virtual processor.* See class file in the figure 2, and translatōr in Figure 2.

As per Claim 3: Kazi further discloses: *The method as claimed in claim 2 in which the class file includes a plurality of methods, and which some or all the methods in the class file are converted to a respective virtual processor tool.* See class file in Figure 2, because class file includes a plurality of methods.

As per Claim 4: Kazi further discloses: *The method as claimed in claim 2 in which the class file includes a call to a method,. and in which the virtual processor code provides a call to a corresponding tool.*
See class file in Figure 2, because class file includes a plurality of methods, and a method is a call.

As per Claim 5: Kazi further discloses: *The method as claimed in claim 2 in which the class file includes a reference to a field, and in which the virtual processor code provides a fixup tool for use in locating the field.* See class file in Figure 2, because class file preprogram per se and it includes code/field, method, method can do any desired function.

As per claim 15: See rationale address in Claim 1 above.

As per claim 16: Regarding, *The method as claimed in any one of claims 1 which includes:*
(d) transmitting the virtual processor code from a server to a second client device; and
(e) translating the virtual processor code into a different native code which uses an instruction set of a second physical processor of the second client device.
(See Figure 2, and same rationale addressed in As per Claim 1).

As per claim 18: Kazi further discloses *A computer system adapted to carry out the method as claimed in claim 16.* (See Figure 2, and same rationale addressed in As per Claim 1).

As per claim 26: Further in view of 2, Kazi further discloses *The method as claimed in any one of claim 2, including verifying the integrity of the class bytecode, and of any external calls,* because it also include "Java Virtual Machine", that has the feature of this claim.

As per claim 27: Further in view of 2, Kazi further discloses *The method as claimed in any one of claim 2 in which the class file is a Java class file* (See Figure 2).

As per claim 28: Further in view of 2, Kazi further discloses *The method as claimed in any one of claim 2 in which the step of translating the program bytecode into virtual processor code is carried out by a first translator program which is itself written in virtual processor code.* (See Figure 2: Bytecode Translator).

As per claim 29: Further in view of 1, Kazi further discloses *The method as claimed in any one claim 1, in which the step of translating the virtual processor code into native code is carried out by a second translator program which is itself written in virtual processor code.* (See Figure 2: IL Compiler).

As per claim 30: Examiner interprets Claim 30 having the same functionality of Claim 1. See rationale addressed in Claim 1.

As per claim 13: Claim 13 has the same functionality as of Claim 1. See rationale addressed in Claim 1 above.

As per claim 14: As set forth in the rationale connecting to the rejection of Claim 1, it is applied to Claim 13; Kazi further discloses a translator that has means, *including binding the translated tools into a task, and executing the task in native code on the physical processor.* See Figure 2, Target processor, where a processor is used for task execution.

As per claim 17: As set forth in the rationale connecting to the rejection of Claim 1, it is applied to Claim 13; Kazi further discloses, *A method as claimed in claim 13 including executing the different native code on the physical processors of different client devices.* See Figure 2, Target processor: it should be noted that each target processor represents to a client.

10. Claims 6-11 are rejected under 35 U.S.C. 103(a) as being unpatentable over Kazi et al, "Techniques for Obtaining High Performance in Java Program", in further view of Miller et al, U.S. Patent No. 6,389,590.

With regards to Claims 6-8, 10:

As per claim 6, Kazi et al does not explicitely address claim 6. In further view of Miller et al, it discloses which the fixup tool is arranged to return a constant fixup value which is representative of the offset of the said field within an object (col. 7 lines 20-30, col. 7 lines 37-46).

As per claim 7, Kazi et al does not explicitely address claim 7. In further view of Miller et al, it discloses linking the virtual processor code and determining the constant fixup value in dependence upon virtual processor code which has been translated from another class file (col. 6 lines 30-36, col. 7 lines 20-46).

As per claim 8, Kazi et al does not explicitely address claim 8. In further view of Miller et al, it discloses which the fixup tool returns a value which is used to patch a method which gets or puts the value of a field (col. 6 lines 30-36, col. 7 lines 20-46).

As per claim 10, Kazi et al does not explicitely address claim 10. In further view of Miller et al, it discloses which the fixup instructions provide instructions as to how the native code can reference another class, or a field or method in another class (Fig. 3, col. 7 lines 21-45 and col. 8 lines 3-I 1).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to include fixup as disclose by Miller into a code translation at back end as disclosed by Kazi just be cause every translator requires to accurately translate code.

With regards to Claims 9, 11:

As per claim 9, Kazi et al does not explicitely address claim 9. In further view of Miller et al, it discloses which the virtual processor code has, included within it at a plurality of points, fixup instructions which indicate that the code at the said points has to be modified by the respective fixup instruction prior to use (Fig. 3, col. 7 lines 21-45 and col. 8 lines 3-11).

As per claim 11, Kazi et al does not explicitely address claim 11. In further view of Miller et al, it discloses which the fixup instructions are transferred, functionally unaltered, by the native translator into the native code (col. 1 lines 15-20, col. 6 lines 1-10 and Fig. 2); the fixup instructions being replaced with native instructions when the native code is bound on the said real physical processor (col. 1 lines 15-20, col. 6 lines 1-10 and Fig. 2).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to include fixup as disclose by Miller into a code translation at back end as disclosed by Kazi just be cause every translator requires to accurately translate code.

Conclusion

11. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 5:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV
March 16, 2007

[illegible]

TED VO
PRIMARY EXAMINER
TECHNOLOGY CENTER 2100